THE OFFICE OF THE STATE CHIEF INFORMATION OFFICER
ENTERPRISE TECHNOLOGY STRATEGIES

North Carolina Statewide Technical Architecture

Data Domain

STATEWIDE TECHNICAL ARCHITECTURE

Data Domain

Initial Release Date:	September 11, 2003	Version:	1.0.0
Revision Approved Date:			
Date of Last Review:	March 17, 2004	Version:	1.0.1
Date Retired:			
Architecture Interdependencies:			
Reviewer Notes: Reviewed and updated office title and copyright date. Added a hyperlink for the ETS email – March 17, 2004.			

Office of Enterprise Technology Strategies
PO Box 17209
Raleigh, North Carolina 27699-7209
Telephone (919) 981-5510
ets@ncmail.com

Data - Data Architecture

Principle 4.00.01 Value and protect the state's data as one of the most critical assets in the organization.

Rationale:

- The successful delivery of government services depends on conclusions derived from accurate, well-maintained, and secure data.
- Protecting the value of the state's data is everyone's responsibility, especially business users and Information Technology staff.
- Leverage and maximize the use of data throughout the state.
- Implementing and enforcing data security is crucial to the organization.

Data - Data Architecture

Principle 4.00.02 Design an adaptive data infrastructure.

Rationale:

- Design the data infrastructure to easily accommodate changes in the data model and database technology. The data infrastructure is a crucial component of establishing an overall adaptive architecture.
- An adaptive data infrastructure provides extensibility in adding new functionality and facilitates vendor independence.

Data - Data Architecture

Principle 4.00.03 Design the enterprise Data Architecture so it increases and facilitates the sharing of data across the enterprise.

Rationale:

- Sharing of data greatly reduces data entry and maintenance efforts.
- Data sharing requires an established infrastructure for widespread data access. This includes integration with the Application, Componentware, Integration, Messaging, Network, and Platform Architectures.
- Consistent shared data definitions ensure data accuracy, integrity, and consistency.
- Data sharing reduces the overall resources required to maintain data across the enterprise.
- For more information, refer to the Federated Metadata topic in this chapter.

Data - Data Architecture

Principle 4.00.04 Design the enterprise Data Architecture so it is business driven, as opposed to technology driven, and aligned with the Application Architecture.

Rationale:

- By designing a Data and Application Architecture that is business driven as opposed to technology driven, an infrastructure is created that is adaptive and flexible to changes in business need.

- Data Architecture supports established business and data manageability requirements.
- Consider both business and application performance drivers.

Data - Data Architecture

Principle 4.00.05 Create provisions for supporting data in a distributed environment.

Rationale:

- In order to maintain a high level of data integrity and data quality, the Data Architecture must simplify the management of distributed data.
- A centralized repository of database-related information, (a.k.a. metadata), facilitates distributed data management.

Data - Data Architecture

Principle 4.00.06 Create and maintain roles and responsibilities within the distributed enterprise Data Architecture to facilitate the management of data. This requires a working relationship between the business user organizations and information services (IS).

Rationale:

The business users are responsible for establishing and maintaining the accuracy of data collected and entered into applications because these systems support the business need.

Business responsibilities are to:

- Provide accurate business definitions of data.
- Develop enterprise-wide business views of shared data.
- Provide business drivers to support centralized data administration.
- Make federated metadata available.
- Define security requirements for data.

IS Responsibilities are to provide a robust technical infrastructure that includes:

- Open, accessible, and adaptable database management systems (DBMSs).
- Centralized data administration.
- Data replication facilities.
- Backup and recovery.
- Security.
- Database monitoring tools.
- Data quality monitoring tools.
- Application mechanisms for helping to ensure accurate data input.

Data - Data Architecture

Principle 4.00.07 Separate the data sources for online transaction processing (OLTP) data and online analytical processing (OLAP) information.

Rationale:

- Separate data sources isolate OLTP systems, which perform mission critical business processing, from large ad hoc queries and online analytical data processing. If the data sources are not separate, ad hoc queries and direct access of data for OLAP systems can adversely impact online transactional processing.
- Data design is adapted for optimal performance for each type of application, OLTP or OLAP. For optimal performance, OLTP and OLAP may require different database designs. OLAP typically includes complex operations involving time series, dimensional, and trend analysis, which do not perform well with relational database technology alone (e.g., sometimes other methods of data storage are needed to support OLAP, such as multi-dimensional databases or flat files). Note: For more information about OLAP, refer to the Data Warehouse Architecture chapter.

Data - Data Architecture

Principle 4.00.08 Information is one of the most valuable assets for making business decisions.

Rationale:

- The successful delivery of government services depends on conclusions derived from accurate, timely, well maintained, and secure information.
- The value of information is proportional to its availability. If information is not accessible or is too hard to use, it is of questionable value.
- Information used frequently or used by many organizations is extremely valuable.

Data - Data Architecture

Principle 4.00.09 The role of the data warehouse is to accelerate the business decision making process.

Rationale:

- New legislation is constantly redefining the services provided by the state. An accelerated decision making process is required, using timely, easily accessible, understandable, reliable, and high quality information.

Data - Data Architecture

Principle 4.00.10 Data warehouse efforts need support from all levels of the business organization, from the business end user to the management level.

Rationale:

- Initial data warehouse investments can be significant, but the return on investment (ROI) is worth it.
- Business end users need to be involved in the data warehouse effort from the beginning.

Data - Data Architecture

Principle 4.00.11 In general, there is no new data, but there is new information. Existing data from multiple sources is being transformed into intelligent and proactive information.

Rationale:

- How the data is used is far more important than the data itself (i.e., even if data is accurate, it can still be used ineffectively).
- The most effective use of data is to turn it into proactive information that responds to business events (e.g., the "push model" of information).

Data - Data Architecture

Principle 4.00.12 Business units are demanding faster access to more information (not just data).

Rationale:

- Data warehouses are most successful when they are built to support business user demands.
- The growing need for data warehouses is an enabling force for implementing enterprise N-tier client/server systems. N-tier client/server systems segregate data from application processes and make it available for use by other applications.

Data - Data Architecture

Principle 4.00.13 Decision-makers should not be overwhelmed with an excessive volume of unnecessary information.

Rationale:

- Too much information gets in the way of focusing on the most important issues that arise at a specific point in time. Some data warehouse efforts put any data in a data warehouse that may be useful in the future, not just the information to assist in a business need. If there is much more information than is needed, it can overwhelm an end user rather than answering their specific decision support need.
- Information that supports OLAP analysis should be proactively presented to business users. Information in a data warehouse can be presented to business end users so that they know it is available and they can use it.

Data - Data Architecture

Principle 4.00.14 A major source of application developer productivity begins in the data warehouse.

Rationale:

- If a data warehouse is properly implemented, end users can perform their own ad hoc queries and reports against the data warehouse. Application developers do not have to develop programs to anticipate and address each reporting need.

Data - Data Architecture

Principle 4.00.15 A data warehouse will continue to evolve over the life of the business.

Rationale:

- Business and information needs are constantly growing and changing, requiring the addition or modification of data stored in a data warehouse.
- Changes in the use of or in the design of the warehouse should be anticipated and expected.

Data - Data Architecture

Principle 4.00.16 Online transaction processing (OLTP) databases and online analytical processing (OLAP) information databases should have separate data storage areas.

Rationale:

- Separate data storage isolates OLTP systems, which perform mission critical business processing, from large ad hoc queries and online analytical data processing. If data storage is not separate, ad hoc queries and direct access of data for OLAP systems can adversely impact online transactional processing.
- Data design for each type of application, OLTP or OLAP, can be optimized for performance. OLTP and OLAP may require different database designs. OLAP typically includes complex operations involving time series and trend analysis, which do not perform well with relational database technology alone (e.g., sometimes other methods of data storage are needed to support OLAP, such as multi-dimensional databases or flat files).

Data - Federated Metadata

Standard 4.01.01 Conform to N.C. GILS standards.

Rationale:

- The U.S. government created a Government Information Locator Service (GILS) to assist the public in locating information from federal agencies. This service is an electronic "card catalog," which describes and indexes information resources of many types.
- Content standards and search and retrieval protocols for US GILS are governed by the "Application Profile for GILS", version 2. The GILS profile establishes a set of attributes for describing and indexing information in a standard way. Further, the profile incorporates portions of the ANSI/NISO Z39.50 Search and Retrieval Protocol (also called ISO23950). Z39.50 is an open communications protocol for searching and retrieving electronic information over a network. Its use enables uniform access to a large number of diverse information sources over the Internet.
- Several states and foreign countries have adopted the GILS standard, making it the foundation of a Global Information Locator Service.
- As a result of Executive Order 100 (September 12, 1996) from Governor Hunt and to meet the needs of agencies to develop diverse types of metadata without duplication of effort,

North Carolina has established a government information locator service compliant with the federal GILS. North Carolina's implementation of GILS is called NC GILS.

- State and local agencies in North Carolina are required by the Public Records Law (North Carolina G.S. § 132-6.1 (b)) to index all databases created or significantly modified. NC GILS provides a standard way for agencies to comply with the amended public records law when indexing their databases. Information about databases indexed through NC GILS will be linked to other indexes developed by the federal government, state and local governments, and foreign countries, thus enabling global exchange of public information.
- A central gateway for searching all NC GILS indexes is available on the Internet at http://www.findnc.net or http://www.findnc.org. For more information about NC GILS, refer to the "Guidelines for Agencies Using the North Carolina Government Information Locator Service (NC GILS)." The guidelines are available on the NC GILS Development Headquarters Web site at http://www.ncgils.state.nc.us/.

Data - Federated Metadata

Best Practice 4.01.01 Use and actively maintain the Federated Metadata Repository to store federated metadata definitions.

Rationale:

- Storing data element definitions in a central repository incrementally builds the enterprise data model.
- The repository must be actively maintained (e.g., changes to metadata occur in the repository before the changes occur in operational applications).
- The repository serves as a centralized data administration tool and helps promote data reusability, reliability, and sharing across the enterprise.

Data - Federated Metadata

Best Practice 4.01.02 When designing or modifying a database, review the Federated Metadata Repository for existing standard and proposed data elements before implementing a new database to ensure data elements are defined according to FMR standards.

Rationale:

Design reviews are essential to ensure that shared statewide data is defined consistently across all applications. Design reviews also determine whether data that already exists is consistently defined and not redundantly stored.

Design reviews should document the following:

- Where is this application getting its data?
- What other applications are getting data from this application?
- Is data used by this application defined consistently with statewide definitions? If not, is there a plan to define the data according to enterprise definitions?
- A design review evaluates the data requirements of a project and identifies the following:
- A data requirement that can be solved by using existing federated metadata element.

- Data not already identified as federated metadata must be proposed as an inter-agency or statewide standard to the Metadata Element Review Team to become federated metadata.
- Access is available for application development projects to reference the FMR in order to actively research data requirements. Review the existing standard and proposed data elements in the FMR before implementing a new database to ensure data elements are defined according to standards.
- Key information about data is stored in the systems that are already implemented in the state. If possible, evaluate existing systems to propose statewide and agency data elements.

Data - Federated Metadata

Standard 4.01.02 Conform to the N.C. Public Records Law.

Rationale:

- Any database that falls under the Public Records Law (North Carolina G.S. § 132-6.1 (b)) must be indexed according to the guidelines established by the N.C. Division of Archives and History and published as Public Database Indexing Guidelines and Recommendations. A copy of this publication can be obtained by calling (919) 733-3540 or at http://www.ah.dcr.state.nc.us/e-records/default.htm.
- For more information regarding the retention and destruction of any electronic records, refer to the web site http://www.ah.dcr.state.nc.us/e-records/default.htm or contact the State Records Center staff at (919) 733-3540.

Data - Federated Metadata

Standard 4.01.03 Populate the Federated Metadata Repository with database information for any project requiring IRMC certification (both custom systems and commercial off-the-shelf systems).

Rationale:

- By submitting the appropriate database information to the FMR, an application will automatically comply with both the NC Public Records Law and NC GILS.
- Established federated definitions facilitate collaboration.
- Refer to: http://www.federated.its.state.nc.us/ for more information about what database-related information must be defined in the Federated Metadata Repository.

Data - Federated Metadata

Best Practice 4.01.03 Define existing databases in the Federated Metadata Repository.

Rationale:

- If possible, existing databases should be defined into the database component of the FMR.
- Centralized data management is crucial to the quality and consistency of shared data and requires a quality assurance and quality control process in place for enterprise data.
- As decentralized databases are implemented across the organization, centralized administration will be

- crucial to the quality and consistency of the data. Use of inaccurate and inconsistent data is of questionable value.

Data - Federated Metadata

Standard 4.01.04 Custom systems must comply with existing FMR standard data element definitions.

Rationale:

- New databases belonging to custom systems must comply with FMR element definitions. The Metadata Element Review Team will review the data elements to determine if the elements conform to existing standards.
- Any new potential data element standards must be proposed and reviewed for approval as a state standard.
- If the data element definition cannot be customized to conform to existing standards, a waiver must be requested.

Data - Federated Metadata

Best Practice 4.01.04 Use the FMR and the Metadata Element Review Team to create federated definitions of enterprise level data and encourage the sharing of data across agencies.

Rationale:

- Data used by multiple business units must be commonly understood and consistently referenced by all business users. Enterprise sharing of data can only be achieved by creating federated definitions.
- Federated definitions of data emerge through the context of projects. An enterprise model evolves over time through ongoing projects.
- In order to create federated definitions of data, cooperation is needed among the business data owners.
- Centralized management of distributed enterprise-level data is provided through the state's Federated Metadata Repository. This repository is available through the Internet.
- The state's Technical Architecture and Project Certification Committee of the IRMC currently approves any statewide and inter-agency standards.

Data - Federated Metadata

Standard 4.01.05 Commericial off-the-shelf (COTS) systems that support client-controlled data element definitions must comply with FMR standard for data element definitions. Or, vendor must provide conversion routine that meets metadata exchange standards for data sharing

Rationale:

- If an off-the-shelf system has data formats that can be modified, the data elements should be adapted to conform to the standard data requirements.

- If the data element definition cannot be customized to conform to existing standards, the vendor must provide conversion routines to conform to the FMR metadata exchange standards.

Data - Federated Metadata

Best Practice 4.01.05 Identify authoritative sources for federated metadata.

Rationale:

- Authoritative business sources for federated metadata must be identified, documented, and actively maintained in the repository. Authoritative business sources are the business units responsible for the accuracy of the data stored.
- A source of record is an authoritative source for data. Data in a source of record is trusted to be accurate and up-to-date. All other data stores should synchronize to the source of record. The data in record sources must be actively managed and the data model should be verified by data administrators. Tools and quality control techniques must be applied to the contents of the data stores themselves, in order to ensure the - quality of the data.
- Each application must identify data sources for all data that it does not originally capture. The application capturing the original data is the authoritative source, and is responsible for the quality of the data. All application data models for ongoing projects should be reviewed to ensure that data existing in authoritative systems is reused and not redundantly stored.

Data - Federated Metadata

Standard 4.01.06 Use Federated Metadata Exchange Standards when exchanging data across agencies.

Rationale:

- If data needs to be exchanged across agency boundaries and the data is physically stored differently, then the data must be exchanged through the exchange standard as specified in the FMR.

Data - Data Modeling

Best Practice 4.02.01 Determine the initial business requirements using terms and tools that are familiar to business users.

Rationale:

- For example, use a simple tool like a spreadsheet containing familiar business terms for the data.
- Once a simple high-level view is designed, data modeling tools can be used internally to develop a more detailed data model.

Data - Data Modeling

Standard 4.02.01 Comply with Federated Metadata Repository standard data element definitions.

Rationale:

- For more information about the Federated Metadata Repository, refer to the Federated Metadata topic in this chapter.

Data - Data Modeling

Best Practice 4.02.02 Take the Entity-Relation (ER) model to the third normal form, then denormalize where necessary for performance.

Rationale:

- The third normal form is the most commonly recommended form for the ER model.
- In some cases, a denormalized database can perform faster as there can be fewer joins, or reduced access to multiple tables. This process saves both physical and logical input and output requirements.

Data - Data Modeling

Best Practice 4.02.03 In a dimensional model, use a star schema whenever possible.

Rationale:

- Use a snowflake schema only if it increases user understandability or improves performance. A snowflake schema may add unnecessary complexity to the data model.

Data - Data Modeling

Best Practice 4.02.04 Restrict free form data entry where possible.

Rationale:

- In the design phase, consider the values that may be input into a field. These values or domains should be normalized so that data is consistent across records or instances. For example, using consistent values for gender or address information.
- Use look-up tables and automate data entry for column or attribute domain values to restrict what is entered in a column.

Data - Data Modeling

Best Practice 4.02.05 Setup indexes and form relationships carefully.

Rationale:

- Limit the number of indexes on databases that will be experiencing significant insert and update activity. When an insert is performed, not only is the record updated, but all the indexes are updated as well.
- Increase the number of indexes on databases where importance lies in retrieval time. Indexes can increase performance on retrieval time.
- Before creating a database, indexes, or data access programs, verify that all relationships have been documented.

Data - Data Modeling

Best Practice 4.02.06 Design the data model to allow for growth and/or change.

Rationale:

- Design data models to accommodate any future changes, including growth and changes in business requirements or database technologies.

Data - Data Modeling

Best Practice 4.02.07 Archive and protect the data model.

Rationale:

- Data models store a wealth of agency and statewide information and must be archived and protected.
- Data models should be catalogued with the agency's project documentation and used to facilitate future revisions.

Data - Data Modeling

Best Practice 4.02.08 Each agency should standardize on a common data modeling tool for designing and maintaining all new database instances.

Rationale:

- Agency Technical Architectures specify each agency's common data modeling tool.
- A data model ensures that data is defined accurately so it is used in the manner intended by both end users and remote applications. For more information about data modeling, refer to the Data Modeling topic in this chapter.

Data - Data Modeling

Best Practice 4.02.09 Use a data modeling tool to reverse engineer existing databases.

Rationale:

- Data modeling tools can evaluate an existing database structure and reverse engineer a data model. The reverse engineered data model can be used to capture valuable information about the existing database.

Data - Data Base Management System

Standard 4.03.01 New databases must use a relational DBMS supporting ANSI-standard SQL (currently SQL92).

Rationale:

- Relational databases offer dependability, flexibility, and compatibility for future data needs.
- Data can be maintained and readily accessed through ANSI-standard SQL calls. SQL is an industry standard for the data access tier of an application and for data access tools.
- Use of proprietary extensions creates vendor lock-in.

- Desktop database products, such as Microsoft Access and FoxPro, are considered end user database access tools and must not be used for agency or statewide implementation.
- Non-relational technology such as flat files can be used for temporary work storage and unstructured data such as textual data. Large-scale use of non-relational technologies must obtain a waiver.
- Use of ODBMS should not be used since the object defines each data type, data is stored in a proprietary format and the data cannot be understood without the object code that describes it. If ODBMS is to be used, a waiver must be requested.

Data - Data Base Management System

Best Practice 4.03.01 When using object-oriented programming languages, use a relational database management system (RDBMS).

Rationale:

- Although there is cost and effort associated with an object-relational model, the relational data model is much more adaptive and understandable. ODBMS only allows access to the encapsulated data defined in the source code. RDBMS allows new relationships and data mappings to be easily defined.
- As applications and associated databases age, applications tend to depreciate because business needs change over time, while data and databases appreciate because of the valuable information contained within.

Data - Data Base Management System

Best Practice 4.03.02 When using a relational database with object-oriented programming, design the relational data model first.

Rationale:

- Object models are typically more complex than the relational model.
- If the object model is built first, building the relational model that matches it may not be possible.
- For more information about relational data modeling, refer to the Data Modeling topic in this chapter.

Data - Data Base Management System

Best Practice 4.03.03 When creating an object-relational mapping between the object model and the RDBMS, keep it simple.

Rationale:

- Simple relationship mappings between objects and relational databases provide ease of use and better performance.
- Use the primary and foreign key relationships in the RDBMS to assist in mapping relationships between objects.

Data - Data Base Management System

Best Practice 4.03.04 Replicate stable data, based on business and performance requirements.

Rationale:

- Replication should not be used unless it is required for performance or decision support.
- A replication infrastructure is simpler to design for stable data. If replicated data is updated frequently (i.e., not stable), it is much more difficult to design and maintain a replication infrastructure.
- For more information about replication, refer to the Data Replication Tools topic in the Data Warehouse Architecture chapter.

Data - Data Access Middleware

Best Practice 4.04.01 Provide centralized administration for data access middleware through central IT staff.

Rationale:

- Typically, workstations and servers are used for multiple applications and require connectivity to multiple databases. If administration for data access middleware is provided by central IT staff, any changes that are required are more easily managed and executed.
- Support costs and efforts to support data access middleware are reduced.

Data - Data Access Middleware

Best Practice 4.04.02 Avoid use of extensions that create vendor lock-in.

Rationale:

- To differentiate their database from other vendors, many database vendors have implemented special extensions beyond the SQL-compliant commands. Although sometimes these extensions may be useful for a particular function, they are not recommended.

Data - Data Access Implementation

Best Practice 4.05.01 Establish a data infrastructure that can accommodate rapid changes in data models based on changes in business requirements or changes in database technologies.

Rationale:

- Business requirements change frequently. The data infrastructure and design must be adaptive and allow for changes to be easily implemented.
- Technology changes are fast emerging. The infrastructure must allow for replacement of the database technology if necessary.

Data - Data Access Implementation

Standard 4.05.01 Use the North Carolina Service Broker (NCSB) for inter-agency data sharing.

Rationale:

- NCSB is the standard for inter-agency data sharing. Inter-agency services deployed using NCSB can be easily leveraged by other authorized applications.
- The agency owning the data is responsible for writing the shared service to access the data. This ensures data integrity and proper data interpretation.
- The agency requesting the data is responsible for writing the request to retrieve shared data according to the shared service specifications.

Data - Data Access Implementation

Best Practice 4.05.02 Centralize data that needs to be shared and current.

Rationale:

- High-volume transaction data that is shared across locations and that needs to be current for all locations must be centralized so all locations have access to the same data source.
- Replicating frequent updates to distributed databases increases systems complexity and network traffic.
- Data must be centralized when one or more of the following criteria occur:
- Many users need access to latest data (i.e., OLTP systems).
- The number of users is small and there are no distributed sites.
- There is a lack of skills and tools at multiple sites to manage distributed data.
- There is a need to provide a consolidated and integrated database for federated metadata on an open platform.

Data - Data Access Implementation

Standard 4.05.02 Use the industry standard of ANSI Standard SQL (currently SQL92) when accessing relational databases.

Rationale:

- When using a database access tool that uses SQL calls, do not use any vendor specific extensions.

Data - Data Access Implementation

Best Practice 4.05.03 Design databases to be modular, business driven and aligned with application services, not monolithic.

Rationale:

- Aligning data with the application service facilitates changes in business processes. Only the data associated with a particular business process is potentially affected when a change is needed, not all the data associated with an entire application. It also increases performance for backup and recovery and provides higher reliability, availability, and scalability.

- By modularizing data, this practice provides better performance for backup and recovery, higher reliability, availability, and scalability, and better transaction performance due to parallelism (e.g., a complex request can be broken down and be processed by multiple databases at the same time).
- Very large databases (VLDBs) typically use a terabyte or more of storage. It is recommended that VLDBs be partitioned based on the appropriate business elements to improve application and database performance. VLDB partitioning can enable more efficient backup and recovery capabilities (e.g., it is more efficient to backup five 10 million row tables simultaneously than it is to backup one 50 million row table).

In order to align data with application services, the following items need to be defined:

- Business processes.
- Data required to service the business processes.
- Business units responsible for providing the business process.
- Access to the data, and the know-how to use the data, by other business units or applications.

Data - Data Access Implementation

Best Practice 4.05.04 Protect data through data access rules.

Rationale:

- There should be no direct access to data. The only access should be through data access rules that own the data. This practice helps to protect data from unauthorized or accidental access.
- For more information on applications, refer to the Application Architecture chapter.
- For more information about data modeling, refer to the Data Modeling topic in this chapter.

Data - Data Access Implementation

Best Practice 4.05.05 Validate data at every practical level to ensure data quality and avoid unnecessary network traffic.

Rationale:

- Validation can be coded into multiple tiers of the n-tier architecture to ensure that only valid data is processed and sent across the network. For example, an invalid field entered in a data entry form can be corrected before data is written to the database.
- Data integrity verification rules should be used when possible.
- For more information about application development, refer to the Application Architecture Domain.

Data - Data Access Implementation

Best Practice 4.05.06 Minimize the replication of data within operational applications by replicating only stable data when necessary and based on business requirements.

Rationale:

- It is better to maintain only one version of data whenever possible, particularly for mission critical OLTP systems.
- Replication must not be used unless it is required for performance or decision support.
- A replication infrastructure is simpler to design for stable data. If replicated data is updated frequently, it is much more difficult to design and maintain a replication infrastructure.
- Replication may be appropriate when there are users in different locations needing similar data that does not need to be current 24 hours a day and a central source database is not a possible solution. (See Figure 4-22.)
- Specific application requirements for data availability, recoverability, and freshness (near real time, 24 hours old, etc.) must be identified.
- Long binary or text value.

Data - Data Access Implementation

Best Practice 4.05.07 Design for all replicated data to be read only.

Rationale:

- Updates must be directed to the authoritative source, not the replicated data.

Data - Data Access Implementation

Best Practice 4.05.08 Perform all data updates to the authoritative sources, then replicate changes to remote databases.

Rationale:

- An authoritative source for data is the source of record where data is collected and maintained by the application that owns the data. All other data stores must synchronize to the source of record. All data updates must occur against the source of record through the data access rules that own that data. (See Figure 4-23.)
- The N-tier Application Architecture facilitates the implementation of reusable data access rules.
- For more information on accessing data stored in legacy systems, refer to the Integration Architecture chapter.
- Long binary or text value.

Data - Data Access Implementation

Best Practice 4.05.09 When replication is needed, evaluate the replication options and select the implementation that meets the existing business needs.

Rationale:

The following considerations must be made:

- Acceptable lag times must be defined to determine replication schemes, schedules, and the product features needed.

- Replication requirements must be defined to determine if the replication tools are sufficient.
- Business requirements must be documented for any data transformation requirements and for any differences in replication requirements.
- How the impact of processing overhead on the source and target databases will affect system performance.
- How network traffic may impact communication costs and performance.
- Tools providing capability for configuration, monitoring, tuning, and administration should be analyzed.

Data - Data Access Implementation

Best Practice 4.05.10 Design the data access infrastructure to support the transparency of the location and access of data by each application.

Rationale:

- This means designing an N-tier architecture where all data access is managed through a middle tier. This design makes databases easy to relocate, restructure, or re-platform the back end services with minimal disruption to the applications that use them. It is essential for adaptive systems.
- For more information on N-tier architecture partitioning, refer to the Conceptual and Application Architecture chapters. For more information on communicating between application tiers, refer to the Application Domain.
- A client should not send SQL requests directly to a server. Instead of using SQL code, the client should communicate with the database through data access rules. The application receives a request from a client and sends a message to the data access rule. The data access rule sends an SQL call to the database. With this method, the client does not send SQL to the server, it sends a request for work. (See Figure 4-24.)
- Long binary or text value.

Data - Data Access Implementation

Best Practice 4.05.11 Design for data to be accessed only by the programs and business rules owning the data, never by direct access to the database.

Rationale:

- This practice ensures security, data integrity and accurate interpretation of the data and allows for adaptability to changes in business needs.
- For more information on accessing the programs that own legacy data, refer to the Application Integration topic of the Integration Architecture chapter.

Data - Data Access Implementation

Best Practice 4.05.12 For data quality management, implement tools, methods, processes and policies to provide high-level data accuracy and consistency across distributed platforms.

Rationale:

- Both business users and Information Technology (IT) staff are responsible for data accuracy and consistency. Policies and procedures must be established to ensure the accuracy of data.
- IT staff is responsible for and must provide security mechanisms to safeguard all data under IT control. The business users must determine functional security requirements, while the physical security must be provided by IT.
- Applied systems management provides safeguards against data loss and corruption and provides the means of recovering data after system failures. This implies that effective backup and recovery systems are imperative and that data can be recovered in a timely basis regardless of the cause of loss.
- To satisfy service-level requirements, if the data is not too volatile, data replication can be used. If the data is extremely volatile, using replicas must be avoided. Additional requirements may include providing redundancy for extra bandwidth for communications volume and for availability in the event of a disaster.
- For critical functions, plan for survivability under both normal operations and degraded operations. (For more information on disaster recovery, refer to the Enterprise Systems Management of IT Assets chapter.)
- It would be ideal to record the flow of data across systems, even if it is built incrementally starting with existing application development projects.

Data - Data Access Implementation

Best Practice 4.05.13 Optimize the physical database design to support an optimal total performance solution.

Rationale:

As with all application and data access design, performance is always a factor to consider. However, database performance is only part of the total solution and must be evaluated in conjunction with other components that impact performance, such as network and application. When implementing data access, several practices can help performance, including:

- Limit the number of indexes in a database. When a record update occurs, not only the record is updated, but all the indexes are updated as well.
- Limit ad hoc data access. End user ad hoc access can impact the performance of the database.
- Limit the number of rows returned in a query. In OLTP, most users normally work with only a single row at a time or a few rows displayed in a grid, list or combo box. If a user will only be working with a handful, there is no reason to return all the rows in a table.
- Return only the columns needed. Provide an explicit column list instead of a "SELECT *" query.
- Limit the number of joins. Complex multi-table joins have negative performance ramifications.
- Avoid sorts. Sorts can be slow, especially if sorting large amounts of data. If sorting is required, sort on indexed fields.

- Limit the rows used for pick lists, combo boxes, or lookup tables. If a large list is necessary, find an alternate method to provide the list.

Data - Data Access Implementation

Best Practice 4.05.14 Implement a minimal number of data access rules.

Rationale:

- A typical n-tier application has numerous business rules. The data access logic for these business rules should be shared through a minimal number of reusable data access rules. Due to the commonality of database queries, many similar queries can execute using a single, properly planned data access routine. (See Figure 4-25.)
- Portability to another database platform or vendor is simplified by having a fewer data access rules.
- Long binary or text value.

Data - Data Access Implementation

Best Practice 4.05.15 Use ANSI-Standard SQL programming language to access a database.

Rationale:

- Data access to relational data stores must be through ANSI-standard SQL programming language access, not proprietary SQL extensions.

Data - Data Access Implementation

Best Practice 4.05.16 Implement a minimal amount of data access rules stored in the database as stored procedures and triggers to avoid vendor lock-in.

Rationale:

- Code data access rules into a data access service. When implemented through the North Carolina Service Broker (NCSB), these services are callable by multiple applications. If a database changes, there is minimal impact to the calling applications since the API should not change.
- Stored procedures and triggers are specific to the database vendor and are more difficult to migrate to a view database if required.
- Database triggers must only be used to support referential integrity.
- Other technologies such as object transaction monitor (OTM) can be used to negate the impact of executing dynamic SQL.

Data - Data Access Implementation

Best Practice 4.05.17 Use the North Carolina Service Broker (NCSB) for intra-agency and intra-application data sharing.

Rationale:

- NCSB is already the standard for inter-agency data sharing. By using the NCSB for intra-agency and intra-application data access, the service can easily be adapted if other authorized applications require data sharing.
- The agency owning the data is responsible for writing the shared service to access the data. This ensures data integrity and proper data interpretation.
- The agency requesting the data is responsible for writing the request to retrieve shared data according to the shared service specifications.

Data - Data Access Implementation

Best Practice 4.05.18 Use the state's interface engine for data sharing of legacy platform data or other data where the application source code cannot be modified or interfaced.

Rationale:

- Some legacy systems do not have an application program interface (API) and there is no access to the source code. When requiring data access to one of these systems, use the state's interface engine.
- The state's interface engine can be used in instances where the source code is unable to be modified or the data layouts are unavailable. It is an unobtrusive interface to accomplish data sharing.
- Use of database gateway or database-specific middleware must be avoided.
- For more information about the state's interface engine, refer to the Integration Architecture.

Data - Data Security

Best Practice 4.06.01 Perform a risk assessment for the application database and data elements to determine level of security required.

Rationale:

- To assure adequate protection of data assets, perform a risk assessment to identify specific security concerns that must be addressed before development and deployment of an application.
- The security analysis will determine what measures must be put in place to restrict end users and applications from viewing, modifying, or deleting confidential or private data. The security analysis may reveal that adequate measures are in place to restrict end users and applications from viewing, modifying, and deleting low impact and public data.
- Classify users according to their functional data needs (e.g., outside access from business partners, citizens, suppliers, etc.)

Data - Data Security

Standard 4.06.01 Change all default database passwords (i.e., sa accounts, etc.).

Rationale:

- System administrator accounts have full access to all databases in a database server. Hackers often attempt a login to a system administrator account using a default password. As soon as a database is set up, change all default passwords.

Data - Data Security

Best Practice 4.06.02 Use generic, protected user accounts for direct database access to streamline administration, ensure scalability, and protect against non-application data access.

Rationale:

- When a generic, protected user account is used, each individual user account is not defined to the database, so end users are unable to gain ad hoc access to the data. Their only access should be through the application.
- The individual user account is only defined at the application level, and does not have to be maintained in more than one place.
- Implementing generic users makes applications scaleable since each process is not tied to a specific user.
- The generic user account and password used to access data in the back end must not be protected and not accessible to end users.

Data - Data Security

Best Practice 4.06.03 Implement data security to allow for changes in technology and business needs.

Rationale:

- Implement security to be a roadblock to unauthorized access, but not a hindrance to access by authorized users. Implement the minimal number of sign-on or authentication processes if possible.
- An adaptable security infrastructure must be implemented to allow for changes in technology, business needs, and reactions to intrusions.
- Monitor ITS and industry security alerts and recommendations. Security tools and techniques are rapidly changing and enhancements are being made. Monitor the industry and ITS recommendations and implement changes to security configurations as needed.

Data - Data Security

Best Practice 4.06.04 Handle sensitive data carefully.

Rationale:

- Confidential or private data must not be stored on a laptop without password protection or encryption. Laptops are vulnerable to loss of data through hackers, thieves, and accidents. Sensitive data must be secured on a database server with proper policies and procedures in place to protect the data.
- Ensure that passwords are encrypted both inside application executables and across the transport layer.

- Password and data encryption in databases and laptops can be provided by third party products.
- A backup and recovery plan for databases and laptops must be in place.

Data - Data Security

Best Practice 4.06.05 Provide measures for laptops to backup their data, like zip drives, etc.

Rationale:

- Only non-sensitive data should be stored on a laptop. If possible, the authoritative source must be on a server, and data should be replicated to the laptop.
- When data is stored on a laptop, provide easy-to-use backup facilities. Implement policies to ensure and automate backup.

Data - Data Security

Best Practice 4.06.06 Record information about users and their connections as they update and delete data. Auditing can determine who updated a record and their connection data.

Rationale:

The information that can be captured by the application includes:

- The user account the user logged in with.
- The TCP/IP address the connected user's workstation.
- The certificate information (if using certificates) about that user.
- The old values that were stored in the record(s) before the modification.
- The new values that were input to the record(s).

Data - Data Security

Best Practice 4.06.07 Implement transaction logging so recovery of original data is possible and protect the transaction log.

Rationale:

- Transaction logging records activity on the database and can be used to roll back a transaction.
- Protect the transaction log through access control and backup. Only the database should be writing to the transaction log. All other access should be read only.
- The transaction log should be located on a separate physical disk if possible. If not possible, use RAID to protect the integrity of the log file.

Data - Data Security

Best Practice 4.06.08 Implement security scanning and intrusion detection at the database level if possible.

Rationale:

- Scan the database and database server for potential weaknesses before they become a problem. Implement any recommendations of the security management tool. For example, a tool may advise to disable FTP services on a database server.
- Monitor the database for possible intrusions. For example, monitor and alert when multiple invalid login attempts occur. Intrusion detection protects the database server from attacks from both sides of the firewall e.g., internal network, WAN, or Internet).
- Audit logins, user account creation, and failed login attempts.

Data - Data Security

Best Practice 4.06.09 Ensure data integrity by securing data movement or data transport.

Rationale:

- When high impact, sensitive data is transported through the LAN, WAN, or Internet, ensure that the data is encrypted and protected from alterations. This can be accomplished through Secured Socket Layers (SSL) or Virtual Private Network (VPN).
- Other types of data must be encrypted and protected if there is a risk of the data being altered.

Data - Data Security

Best Practice 4.06.10 Protect database servers from hardware failures and physical OS attacks.

Rationale:

- Database servers must be located in a climate-controlled, restricted-access facility, and preferably a fully staffed data center. Uninterruptible power supplies (UPSs), redundant disks, fans, and power supplies must be used.
- For more information about hardware, refer to the Platform Domain.

Data - Data Security

Best Practice 4.06.11 Protect source code in data access rules, particularly if it contains password information.

Rationale:

- On the back end, an application needs to store account and password information in order to authenticate to a database or other application service. Protect the source code from unauthorized viewing.
- Store passwords in an encrypted format when possible.

Data - Data Security

Best Practice 4.06.12 Do not store credit card numbers in the database for non-recurring charges or infrequent recurring charges. Store authorization numbers and discard credit card numbers after use.

Rationale:

- For infrequent recurring charges (for example an annual fee), or non-recurring charges (for example a one- time fee), storing credit card numbers and expiration dates in a database, even encrypted, can present an unjustifiable risk for the state.
- A credit card number is only necessary to request authorization. Keep the credit card number only until authorization is complete, then discard. The authorization number can be used to track activity and verify authorization.

Data - Data Security

Best Practice 4.06.13 Protect and encrypt credit card numbers when storing for recurring charges. Store personal verification information independently.

Rationale:

- Certain business requirements, such as frequent recurring charges, may require credit card numbers to be stored in a database.
- When it is absolutely necessary to store credit card numbers, encrypt the credit card number in the database. To further protect the credit card, store personal verification information, such as name and address, in a separate database from credit card information. Use different user accounts for each database connection.

Data - Data Warehouse

Best Practice 4.07.01 Do not wait for an enterprise information model to start data warehouse efforts.

Rationale:

- Enterprise models often take years to complete. Meanwhile, business requirements are changing rapidly. Data warehouse efforts begin with a strategic set of information, preferably targeting a cross section of users.
- Develop the data warehouse incrementally:
- Start small. Start by putting a relatively small amount of strategic data on a separate server, solving the problems of a specific set of users. The first step should be measured in time rather than gigabytes. The first stage should be no more than 90 days, from initiation to implementation.
- Once the data is ready, it is time to add on-line analytical processing (OLAP) capability. For complex decision support, add on-line analytical processing, including trend analysis, indexing technology, and multi-dimensional database technology.
- Implement near-real time production data feeds to keep the database populated.
- Apply data extraction technology to accomplish real time feeds to maintain the data. (Note: For real time data feeds, the multidimensional indexing engine must be a separate database because it cannot accommodate real-time feeds).

Data - Data Warehouse

Best Practice 4.07.02 Begin data warehouse efforts by addressing a specific requirement for a specific decision support application, keeping growth and scalability in mind.

Rationale:

- This practice is similar to data mart design, but the tools and databases used should be designed to support a large data warehouse.
- Use vendor supplied products designed to support a large data warehouse. Vendor-supplied data mart tools are not typically scaleable to support the migration from a data mart to a data warehouse solution. These tools are designed to quickly implement a specific solution.

Data - Data Warehouse

Best Practice 4.07.03 Identify specific requirements for data availability, freshness (i.e., live, 24 hours old, etc.), and recoverability.

Rationale:

- Some data warehouses need to be updated more frequently than others. When the original data is frequently changing or is more volatile, it may be necessary to update the data warehouse on a near real time basis.
- On the other hand, if the original data is fairly stable and not as volatile, the data warehouse may only need daily, weekly, or even monthly updates. For example, a data warehouse that stores criminal data contains more volatile information and needs to be updated more frequently than a data warehouse that stores state registered corporation name and address information for public access.

Data - Data Warehouse

Best Practice 4.07.04 Perform benchmarks on the database design before constructing the database.

Rationale:

- Expect to make changes and adjustments throughout development.
- Changes during the early cycles up to, and including implementation, are a primary mechanism of performance tuning.

Data - Data Warehouse

Best Practice 4.07.05 Normalize application access to all decision sources.

Rationale:

- Common functions for data access, such as a "Compute Age" function, should be reusable and shareable by multiple application systems.
- For more information about shared business rules and components, refer to the Application and Componentware Architecture chapters.
- Long binary or text value.

Data - Data Warehouse

Best Practice 4.07.06 Ensure that the appropriate security is applied to the data warehouse.

Rationale:

- Prevent unauthorized users from tampering with the data.
- Control access to portions of the database on a user-by-user basis. Certain types of data that is stored in a data warehouse may be sensitive. Providing protection for privacy of individual and confidentiality of data must be considered.

Data - Data Warehouse

Best Practice 4.07.07 Choose a data warehouse project manager who is user-oriented rather than technology oriented.

Rationale:

- A user-oriented manager ensures that the data warehouse will meet the business needs of the end users.
- The data warehouse project manager must manage the expectations and sponsorship of the data warehouse.
- The data warehouse manager can make sure the data is easy to use and understand.

Data - Data Warehouse

Best Practice 4.07.08 Allow only read only access to end users of data warehouses.

Rationale:

- Updates should only occur to the operational (OLTP) source where the data originates.

Data - Data Warehouse

Best Practice 4.07.09 Direct all information queries against decision support databases, not OLTP databases. Conversely, operational transactions should be directed to operational databases only, not OLAP databases.

Rationale:

- Data warehouses, and data marts contain data that has been checked for consistency and integrity, and represents a cross-functional view of data.
- Data in transaction (OLTP) systems typically support a specific business group or function.
- OLTP transactions should not depend on a data warehouse database. They require a stable operational environment that is not affected by ad hoc usage or external data.

Data - Data Warehouse

Best Practice 4.07.10 Establish the data warehouse as the authoritative source for all other decision support databases.

Rationale:

- Data administration policies, procedures and tools are required.
- Project design reviews are required. A mandatory deliverable must be identification of all data sources for the project.
- Distributed data warehouse servers and data marts should use an authoritative source for data feeds.

Data - Data Warehouse

Best Practice 4.07.11 Store atomic-level data in the data warehouse in addition to summary data.

Rationale:

- Atomic data is transaction-level data. It contains much more detail than summary data.
- Atomic-level data addresses the business need to recast history. Due to the fast pace of business change, many organizations are going through multiple reorganizations. After a reorganization, many decision makers want to recast history (e.g., to get a feel for what test scores would have been like if the number of school districts was already reduced to respond to legislation or funding).
- If only summary level historical data is kept in the data warehouse, it is not possible to recast history.

Data - Data Warehouse

Best Practice 4.07.12 Perform periodic validity audits against the data warehouse information model to ensure a high level of confidence in the quality and integrity of the data.

Rationale:

- Accelerated decision making requires high quality data. If operational data has changed or additional data is needed, changes must be made in the information model and in the data warehouse itself.
- The data stored in a data warehouse should conform to the information model.
- The source data populating a data warehouse should be verified for consistency and accuracy.
- The data warehouse should still correspond to business needs.
- Ensuring the integrity and quality of data is the responsibility of both the business users and IS.

Data - Data Warehouse

Best Practice 4.07.13 The implementation plan should match critical business needs.

Rationale:

- Start with strategic business needs.
- Optimize critical data access before less-critical data access.

- Optimize high-volume data access before low-volume data access.
- Optimize applications and data with high-service-level requirements before those with lower requirements.

Data - Data Warehouse

Best Practice 4.07.14 Consider the network when designing OLAP systems.

Rationale:

- The network is a partner to an application and can impact performance and design.
- Estimate the impact on the network when partitioning data.
- For more information about networks, refer to the Network Architecture chapter.

Data - Data Warehouse

Best Practice 4.07.15 Plan and budget for the ongoing operations, administration, and maintenance of a data warehouse.

Rationale:

- A support plan for the data warehouse should be documented and implemented.
- The data warehouse should be scaleable to meet future demands.

Data - Repository

Best Practice 4.08.01 Maintain a repository for every data warehouse.

Rationale:

- The repository contains metadata, or information about the data, in the data warehouse.
- The repository represents the shared understanding of the organization's data.
- The repository can be built incrementally, in stages, based on data warehouse design and implementation.
- The repository should support multiple types of data elements, such as graphics.

Data - Repository

Best Practice 4.08.02 Actively maintain the repository.

Rationale:

- Changes in the repository must occur before the changes to the data warehouse environment.

Data - Repository

Best Practice 4.08.03 The repository management system used to store metadata for a data warehouse should be built with the same repository tools that are used to manage federated data metadata.

Rationale:

- Federated data metadata reference many common data elements used by multiple application systems. Due to the commonality of federated data, it will frequently be used in a data warehouse environment.
- If the repository databases use the same management systems, metadata will easily transfer between repositories to build the information model.
- For more information about the federated data information repository, refer to the Data Architecture chapter.

Data - Data Hygiene Tools

Best Practice 4.09.01 Use the data warehouse metadata repository to document the rules applying to data scrubbing.

Rationale:

- The information about how the data is to be scrubbed should be saved for historical purposes.

Data - Data Hygiene Tools

Best Practice 4.09.02 Determine the schedule for data cleansing during the design of the data warehouse.

Rationale:

- Depending on the sources of the original data, some data warehouses may need to be cleansed more frequently than others.

Data - Data Hygiene Tools

Best Practice 4.09.03 Ensure data entry quality is built into new and existing application systems to reduce the risk of inaccurate or misleading data in OLTP systems and to reduce the need for data hygiene.

Rationale:

- Provide well-designed data-entry services that are easy to use (e.g., a GUI front end with selection lists for standard data elements like text descriptions, product numbers, etc.).
- The services should also restrict the values of common elements to conform to data hygiene rules.
- The system should be designed to reject invalid data elements and to assist the end user in correcting the entry.
- All updates to an authoritative source OLTP database should occur using the business rules that own the data, not by direct access to the database.
- Attention to detail should be recognized and rewarded.

Data - Data Extraction and Transformation Tools

Best Practice 4.10.01 During data warehouse design, determine the logic needed to convert the data, plan and generate the extraction and transformation routines, and quality assure the data populating the data warehouse.

Rationale:

- Planning for data extraction and transformation should start at the same time the data warehouse design starts.
- Data extraction and transformation is an important process for populating the data in a data warehouse and for ensuring that the data in a data warehouse is accurate.
- Data extraction and transformation logic includes data conversion requirements and the flow of data from the source operational database to the data warehouse.

Data - Data Extraction and Transformation Tools

Best Practice 4.10.02 Assess the source data that will populate a data warehouse for accuracy and quality.

Rationale:

- Data needs to be accurate to ensure good business decisions.
- Data needs to be relevant to the business need and consistent across multiple sources.
- Data must be complete. It must contain the information necessary to answer the data warehouse business need.
- The data assessment also involves evaluating the business rules associated with that data. The appropriate business rules must be applied to the data to maintain accuracy.

Data - Data Extraction and Transformation Tools

Best Practice 4.10.03 Apply data hygiene techniques to the warehouse data after it has been extracted.

Rationale:

- The data warehouse will contain data from disparate operational data sources. This data will need to be cleaned to resolve any differences before it can be stored for analysis in the data warehouse.

Data - Data Extraction and Transformation Tools

Best Practice 4.10.04 Develop a schedule for data extraction that both meets the needs of the data warehouse users and does not impact an OLTP system.

Rationale:

- Evaluate the impact of data extraction to any OLTP systems accessed.

Data - Data Extraction and Transformation Tools

Best Practice 4.10.05 Document data extraction and transformation information in the data warehouse repository.

Rationale:

- Data extraction and transformation metadata are important aspects of a data warehouse. This metadata provides the information map connecting the data populating a data warehouse with its source operational databases.

Data - Data Extraction and Transformation Tools

Best Practice 4.10.06 If a vendor-supplied extraction and transformation product is selected, it should support the same metadata repository that supports the data warehouse. It should also support the physical data warehouse.

Rationale:

- Select products that are capable of interacting with the metadata repository and the data warehouse.
- Metadata drives the operations of the extraction and transformation tools. If the data warehouse repository is not supported by a vendor-supplied extraction and transformation product, a separate metadata repository must be developed and maintained.

Data - Data Replication Tools

Best Practice 4.11.01 Replicated data should be read-only, except where business practices clearly allow inconsistencies.

Rationale:

- It is easiest to manage data quality and integrity when replicated and distributed data is read only.
- Some business applications require updates to occur against the local database.
- Distributed independent updates require a reconciliation process that may be quite complex.

Data - Data Replication Tools

Best Practice 4.11.02 Replicate stable data, based on business and performance requirements.

Rationale:

- Replication should not be used unless it is required for performance or decision support.
- A replication infrastructure is simpler to design for stable data. If replicated data is updated frequently (i.e., not stable), it is much more difficult to design and maintain a replication infrastructure.

Data - Data Replication Tools

Best Practice 4.11.03 When replication is needed, evaluate the replication options and select the implementation that meets the existing business needs.

Rationale:

- Define the acceptable lag times to determine replication schemes, schedules, and the product features needed.
- Document the business needs for any non-identical replicated copies of data and any data transformation requirements.
- Decide if the replication required is within the capabilities of an existing replication product or determine if external processing is required.
- Determine if replication processing overhead on the source and target databases will affect the performance of the applications using either database.
- Determine if network traffic required for replication can impact communication costs and performance.
- Determine the capability of the tools selected for configuration, monitoring, tuning, and administration.

Data - Business Intelligence Tools

Standard 4.12.01 When accessing relational databases, use the industry standard of ANSI Standard SQL (currently SQL92).

Rationale:

- When using a database access tool that uses SQL calls, do not use any vendor specific extensions.

Data - Business Intelligence Tools

Best Practice 4.12.01 Implement as few database access tools as possible.

Rationale:

- If vendor or business specific database access tools are selected that cannot be used to access databases distributed statewide, then multiple tools must be implemented, supported, and maintained.
- Select a tool that has a high degree of functionality, but is easy to learn and use. The learning curve for end users is reduced if they do not have to learn multiple tools.
- Avoid developing custom applications to perform routine data access functions.
- "Common data services" and "shared data" will increase in coming years.
- Support costs and efforts to support database access tools are reduced.

Data - Business Intelligence Tools

Standard 4.12.02 Use ODBC from any data access programs rather than vendor-specific database access tools.

Rationale:

- ODBC allows flexibility in programming. A database can be easily modified or relocated. If a change is needed, the change is made to the ODBC configuration file, not to each business intelligence program or tool.

Data - Business Intelligence Tools

Best Practice 4.12.02 Implement decision support and executive information applications using an N-tier application architecture.

Rationale:

- By developing an application system in N tiers, systems can respond quickly to changes in business needs.
- Decision support and executive information systems application programs benefit from the use of N-tier and reusable and shared components.
- For more information about N-tier application programs and reusable components, refer to the Application Architecture and Componentware chapters.

Data - Business Intelligence Tools

Best Practice 4.12.03 There should be no ad hoc query access to OLTP databases.

Rationale:

- Ad hoc query access to online operational databases can severely impact the performance of mission critical operations. A data warehouse should be implemented for users with ad hoc query needs.

Data - Business Intelligence Tools

Standard 4.12.03 Implement a server-based ODBC solution rather than a workstation-based ODBC implementation.

Rationale:

- A server-based ODBC solution is easier to administer. ODBC database changes and additions are easier to manage, since updates are made to ODBC servers, not every workstation that uses ODBC.

Data - Business Intelligence Tools

Standard 4.12.04 Use domain name system (DNS) names for databases that are accessible via TCP/IP.

Rationale:

- A DNS server provides the capability for a long or complicated TCP/IP location to be accessed by a generic, short alphabetic name. It is basically a lookup service. It maps the generic alphabetic DNS name to its complicated TCP/IP location. The client application programs can be configured to use the generic names when they need to access a database (e.g., a database can be accessed by a client by using the generic name "Summary." The DNS server accepts "Summary" and translates the address into \\UX00001\SRV1\DBASE\DATAWAR.FIL. The client then is able to access the database).
- If the database location changes, the DNS configuration is changed, and no changes are needed to each client configuration.
- Long binary or text value